محمد امانلو تکلیف اول درس پایگاه داده، استاد شاکریَ 810100084

سوال1) الف

نرم افزارهای مدیریت پایگاه داده به منظور ذخیره کردن و مدیریت پایگاه داده‌ها ایجاد شده‌اند. DBMS تمرکز خود را بر روی طراحی منطقی پایگاه داده قرار داده و به کاربر اجازه می‌دهد تا به جزئیات فنی مربوط به ذخیره، بازیابی و شاخص‌بندی داده‌ها فکر نکند. از طرف دیگر، تلاش دارد با ایجاد محیطی امن و مدیریت یکنواخت بر داده، امکان دسترسی همزمان به داده‌ها بدون خرابی و استقلال داده را برای کاربران فراهم کند.
مزایای اصلی استفاده از آن:

1. دسترسی همزمان و بدون ایجاد خرابی در داده ها و قابلیت بازیابی مناسب از crash دارد
2. افزایش سرعت دسترسی و در نتیجه کم کردن زمان ایجاد برنامه ها
3. به دلیل ساختار لایه لایه برنامه های سطح کاربر از جزئیات پیاده سازی و ذخیره سازی داده ها مطلع نیستند، که این باعث می شود داده ها مستقل از یکدیگر باشند.
4. افزایش قدرت مدیریت دسترسی ها، اعمال محدودیت ها و در نتیجه افزایش امنیت داده ها
5. قابلیت مدیریت مرکزی و یکنواخت روی داده ها را به ما می دهد

ب)

وظایف یک ادمین پایگاه داده :

1. طراحی شمای فیزیکی و منطقی مناسب و اطمینان از یکپارچگی آن

2. مدیریت بخش امنیت داده ها و احراز هویت های مناسب برای دسترسی به داده ها
3. ایجاد دسترسی به داده ها و مسیر بازیابی پس از خطا یا crash که در واقع در پی حفظ ترتیب و انطباق اولیه داده هاست
4. آمادگی در برابر تغییر و انجام تغییرات در طول زمان در صورت نیاز با بهینه سازی کوئری ها، تنظیمات سرور و به روز رسانی نرم افزارهای مختلف مورد نیاز

پ)

به دلیل آنکه در برنامه های کاربری مختلف ما نیازمند آن هستیم که برنامه ها را به طور موازی و هم روند اجرا کنیم تا سرعت اجرای آن ها را افزایش دهیم چرا که دسترسی به دیسک به نسبت بسیار کندتر است. به همین دلیل در مجموعه داده ها ممکن است به طور همزمان تغییراتی ایجاد شود. در واقع در این حالت وقتی دو برنامه کاربری به یک مجموعه دادگان دسترسی همزمان دارند وضعیت Race رخ می دهد که ممکن است داده ای که یکی به آن دسترسی دارد داده ای باشد که توسط دیگری تغییر پیدا کرده و یا مثلا حذف شده باشد و در واقع Data inconsistency ایجاد شده است. ما مسئول آن هستیم که این نگرانی را از روی کاربر برداشته و آن را مدیریت کنیم به همین دلیل در اینجا باید از Concurrency control بهره ببریم. در این روش ما بر روی ثبات و استقلال داده ها ، یکپارچگی داده ها و ایزوله بودن آن ها محافظت هایی انجام می دهیم تا این موارد را در داده های خود از دست ندهیم. در این صورت کاربر نهایی می تواند همواره فرض کند که به تنهایی به دادگان دسترسی دارد و هیچکس دیگری مزاحم عملکرد او نیست.

ج) در واقع زمانی که ما می خواهیم یک سری صفات توصیفی مختص به یک مجموعه موجودیت های مشخص را فقط برای برخی از موجودیتهای موجود در آن تعریف کنیم، از سلسله مراتب ISA استفاده میکنیم. در ISA ما مجموعه موجودیت ها را به کمک سلسله مراتب به چندین زیر مجموعه از موجودیت ها تقسیم میکنیم و صفات و روابط

دلخواه خود را برای آنها ایجاد می کنیم، در واقع رابطه ISA نشان دهنده ارث بری یک یا چند موجودیت فرزند از یک موجودیت پدر است. یعنی علاوه بر ویژگی های فرزند تمامی ویژگی های پدر را هم به ارث برده است. به طور کلی دلایل زیر مهمترین دلایلی هستند که به دلیل آن ها ما از ISA استفاده می کنیم:
1. برای افزودن یک ویژگی به یک زیر زیر کلاس که این زیر کلاس ها ویژگی پدرشان را به ارث میبرند.
2. برای تعیین Entity هایی که در یک رابطه وجود دارند و شفاف سازی آن ها در روابط خود

د)

Overlap constraint به این معنای این است که آیا دو زیرکلاس مختلف می توانند یک موجودیت مشترک داشته باشند یا نه
به عبارت دیگر، آیا دو زیرکلاس می توانند همپوشانی داشته باشند یا خیر. به عنوان مثال، یک دانشجو می تواند همزمان هم دانشجوی لیسانس باشد و هم دانشجوی ارشد یا باید فقط یکی از این دو نقش را داشته باشد.

Covering constraint تعیین می کند که آیا موجودیت‌های موجود در زیرکلاس‌ها باید تمام موجودیت‌های سوپرکلاس را شامل شوند یا نه. به عبارت دیگر، آیا این زیرکلاس‌ها باید تمام موجودیت‌های سوپر کلاس را پوشش دهند یا نه. برای مثال، آیا یک کارمند می تواند هم نه کارمند ساعتی باشد و نه قراردادی؟ بله، باید حداقل در یکی از این دو نقش باشد. در واقع در این مثال اگر overlap constraint وجود داشته باشد، هر کارمند نمیتواند همزمان هم کارمند قراردادی باشد و هم کارمند ساعتی و فقط باید در یکی از این دو entity باشد. ولی اگر covering constraint وجود داشته باشد یعنی یک کارمند نمی تواند در entity set محصول قرار بگیرد و حتما باید در یکی از زیرکلاس ها باشد.

سوال 2)

در صورتی که شماره دانشجویی را بعنوان یک کلید اصلی در نظر بگیریم بایستی آن را بعنوان یه ویژگی برای دانشجو لحاظ کنیم تا یکتایی آن حفظ شود. اما در صورتی ما شماره دانشجویی را به عنوان یک موجودیت مستقل تعریف میکنیم که یا شماره دانشجویی هر دانشجو یک چیز ثابت نباشد و در زمان های مختلف متفاوت باشد که مثلا می تواند دو نفر در دو رشته مختلف یک شماره دانشجویی یکسان داشته باشند، در این صورت ما می توانیم یک تاریخچه از شماره دانشجویی ها داشته باشیم. حالت دیگر هم مثلا این است که در شماره دانشجویی هر بخش از شماره دانشجویی معنایی مستقل داشته که نشان دهنده یک ویژگی از دانشجو است در این صورت نیز می توان شماره دانشجویی را بعنوان یک مجموعه موجودیت مجزا در نظر داشت. در ذخیره داده های اگر اطلاعات دانشجو در هنگام دریافت از کاربر یا ثبت نام ترم را بخواهیم داشته باشیم بهتر است که شماره دانشجویی را یک ویژگی در نظر بگیریم ولی مثلا به هنگام حذف و اضافه یا اخذ دروس مختلف اینکه شماره دانشجویی را یک موجودیت در نظر بگیریم می تواند مفید باشد. به طور کلی در صورتی که ما به دنبال داشتن تارخچه شماره دانشجویی ها باشیم موجودیت گرفتن ان کار خوبی است اما اگر به دنبال استخراج اطلاعاتی از شماره دانشجویی هستیم ویژگی در نظر گرفتن ان کار خوبی است

سوال 3)

الف)

محدودیت حداقل یک یا participation constraint: یک موجودیت مجبور به شرکت در حداقل یک رابطه از relationship set باشد.

محدودیت حداکثر یک یا key constraint : یک موجودیت مجبور به شرکت در حداکثر یک رابطه از relationship set مربوطه باشد.

محدودیت دقیقا یک که ترکیب همان key constraint و participation constraint:
یک entity مجبور باشد دقیقا یک رابطه از relationship set مربوطه شرکت کند.

ب)

1_هر کالای ارسالی، توسط حداقل یک خرده فروشی ارسال می شود.

2_ هر خرده فروشی،دقیقا یک کالا را ارسال می کند.

3_هر رویداد حمل و نقل،منجر به ارسال دقیقا یک کالا می شود.

4_هر کالا حداقل از طریق یکی از رویداد های حمل و نقلی،ارسال می شود.

برای تصحیح روابط داریم:

رابطه ی 1، در این رابطه در نظر گرفتن محدودیت غیر منطقی است زیرا یک خرده فروش می تواند کالا های مختلفی را بفروشد.

رابطه ی 2،به این دلیل که یک کالا مشخص باید دقیقا از یک خرده فروش دریافت شود بنابراین در نظر گرفتن محدودیت برای آن غیر منطقی است.

رابطه ی 3،باید محدودیت های را به حداقل یک تغییر دهیم زیرا با هر رویداد حمل و نقلی باید بتوانیم بیشتر از یک کالا ارسال کنیم.

پ)

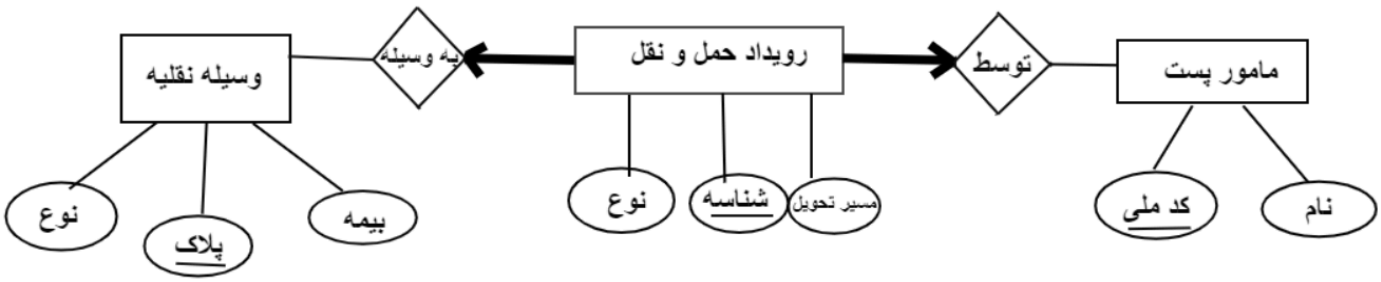
با در نظر گرفتن تغییرات اعمال شده،برای اینکه یک خرده فروش بتواند تاریخچه رویداد های انتقالی تا تحویل هر مرسوله را نگه دارد باید زمان تمام رابطه های مربوط به آن

محصول را پیدا کند و با تعریف رابطه در aggregation مربوطه،خرده فروشی اطلاعات را بدست آورد.

ج)

یکی از ویژگی های کالاهای ارسالی توسط خرده فروش،تاریخ نهایی تحویل است. بنابراین با احتساب اینکه این زمان، همان زمان رسیدن کالا به دست مشتری است و با مقایسه ی این زمان با تاریخی که به مشتری اعلام شده می توان به این نتیجه رسید که تحویل در زمان مقرر انجام شده یا نه.

د)



سوال 4)

الف)

ویژگی ترکیبی ویژگی است که از ترکیب چندین ویژگی دیگر تشکیل شده است.ویژگی تاریخ تولید در این نمودار یک ویژگی ترکیبی است.چون شامل روز و ماه و سال تولید است. میتوان این ویژگی را یک موجودیت در نظر گرفت که با نوبت تولید ارتباط دارد. البته که این ویژگی ها به صورت مجزا اطلاعات کافی برای تعریف یک تاریخ را ندارند و باید کنار هم قرار داشته باشند، اگر اینطور باشد اگر پردازش خاصی نیاز باشد که روی

تاریخ های تولید مربوط به روز یا ماه یا سال قابل انجام باشد به این شکل که گفته شده قابل انجام است.

ب)

ویژگی هزینه تولید در نمودار داده شده را می توان یک derived attribute از ویژگی قیمت هر واحد به حساب آورد. که از ویژگی های دیگر نمودار، ارتباطات آن بدست می آید. هزینه تولید در هر نوبت تولید برابر با مجموع قیمت هر واحد برای تمام مواد اولیه استفاده شده در آن نوبت تولید است.

ج)

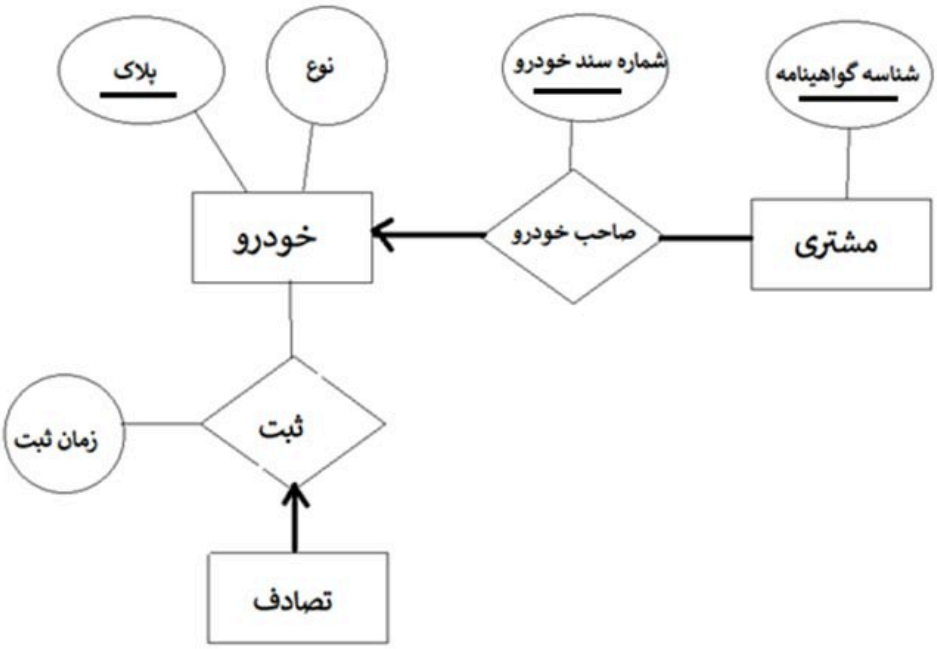
ایجاد Entity به نام "ویژگی محصول" کمک می‌کند تا از تکرار اطلاعات درباره ویژگی‌های مشترک محصولات جلوگیری شود. با تعریف این Entity و اضافه کردن ویژگی‌های مورد نیاز (نوع، وزن، توضیحات) به آن، می‌توانیم این اطلاعات را به صورت ساختاری و یکسان ذخیره کرده و از تکرار ناخواسته آنها در هر محصول جلوگیری کنیم.با تعیین نوع به عنوان primary key برای "ویژگی محصول"، مطمئن می‌شویم که هر ویژگی تنها یکبار و با یک مقدار منحصر به فرد ذخیره خواهد شد. ایجاد یک relation بین "محصول" و "ویژگی محصول" نیز به ما کمک می‌کند تا این دو موجودیت را به یکدیگر مرتبط کنیم و به راحتی به اطلاعات مربوط به هر محصول دسترسی پیدا کنیم.بنابراین، با استفاده از این رویکرد، می‌توانیم از ساختاردهی منطقی‌تر و بهینه‌تری برای اطلاعات مربوط به محصولات استفاده کنیم و از ایجاد اطلاعات تکراری جلوگیری کرده و توسعه و تغییرات در آینده را به شکل ساده‌تر واقعی‌تر کنیم.

د)

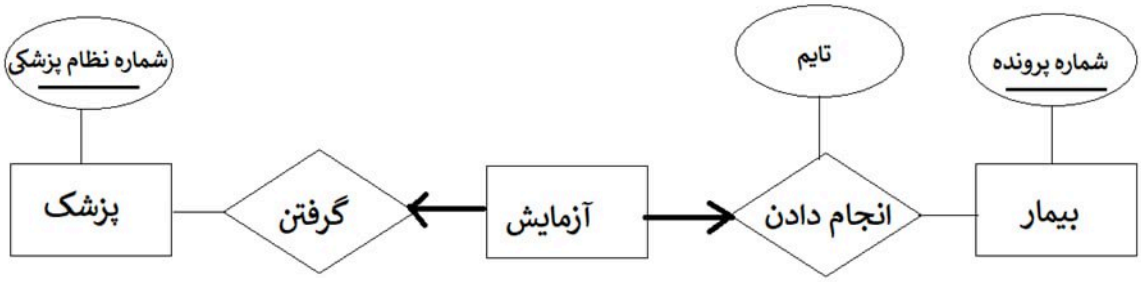
در اینجا فقط ماده های اولیه ای در دیتابیس ما ذخیره میشوند که دست کم در یک نوبت تولید، استفاده شوند پس در صورتی که ماده ای در هیچ نوبت تولیدی مصرف نشده باشد، ما آن ماده را ذخیره نمی کنیم. در واقع چون با این کار هر ماده اولیه ما باید حتما دست کم در یک نوبت تولید استفاده شود پس هیچ گاه امکان انبار کردن ماده

اولیه را نداریم چون انبار کردن بدین معنی است که در هیچ نوبت تولیدی از ماده اولیه استفاده نمیشود. در نتیجه می توان گفت که ماده اولیه به محض در دسترس قرار گرفتن باید در نوبت یا نوبت های تولید مورد استفاده قرار بگیرد.

سوال 5)



سوال 6)



سوال 7)

مفروضات:

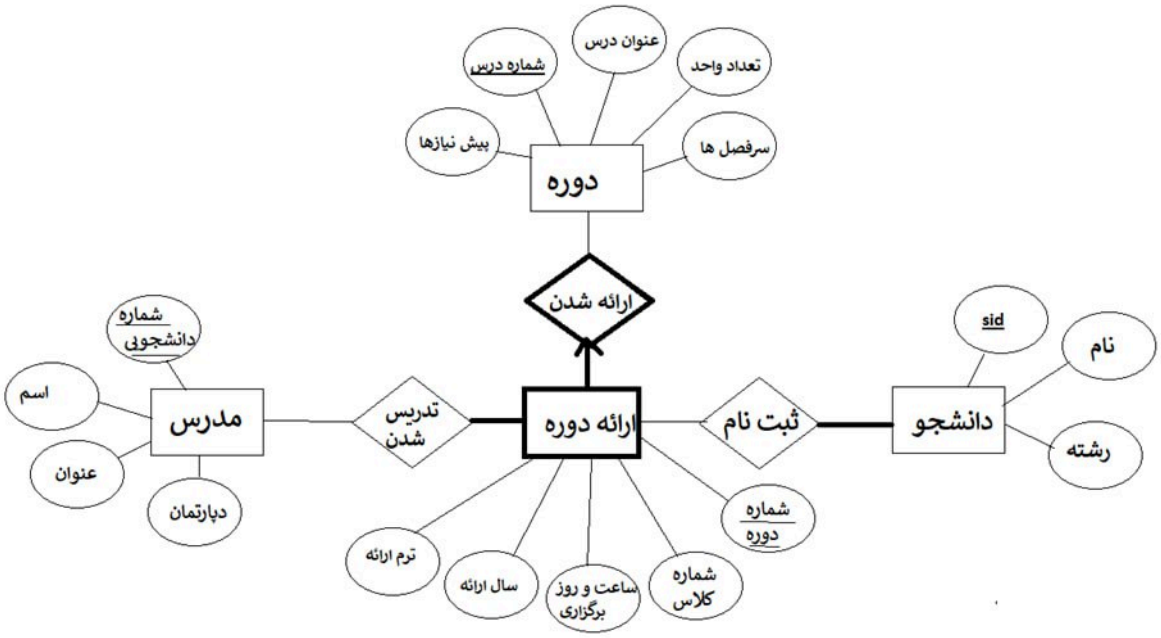
1. هر ارائه دوره دقیقا باید شامل یک دوره باشد

2. به جای اینکه نام مدرس ها را به عنوان یک ویژگی در ارائه که redundancy را زیاد تر می کند آن را به صورت یک رابطه بین ارائه دوره و مدرس بگیریم در نتیجه هر ارائه دوره باید حتما یک مدرس داشته باشد .

3. از دانشجویان هم می توانند ارائه دوره ها را اخذ کنند

4.ممکن است هیچ دانشجویی یک ارائه دوره را اخذ نکند

برای کاهش redundancy پیش نیاز ها را بصورت رابطه نگه می داریم.



سوال 8)

مفروضات:

1. هر تیم میتواند یک یا بیشتر محصول داشته باشد ولی هر محصول متعلق به دقیقا یک تیم است.

2. برای هر تیم یک ناظر محصول وجود دارد که هر تیمی ناظر ندارد، پس هر تیم، حداکثر یک ناظر محصول دارد.

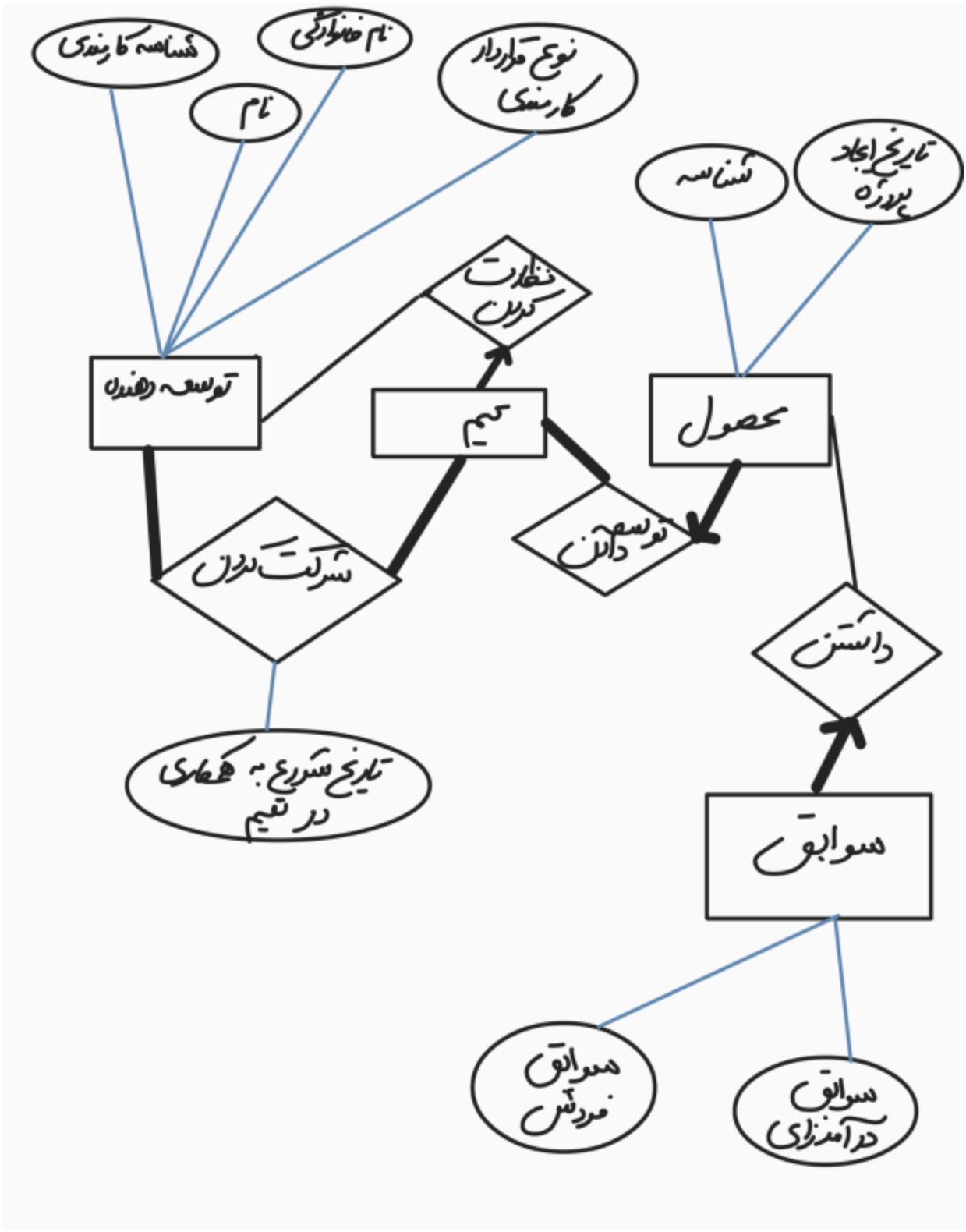
3. شرکت به دلیل عدم ظرفیت کافی عضو جدید نمیپذیرد و ناظر و توسعه دهنده، از بین کارکنان که از موجودیت توسعه دهنده هستند، انتخاب میشوند

4. به علت عدم ظرفیت کافی برای نظیر کردن هر تیم به یک ناظر محصول، هر تیم حداکثر یک ناظر و هر ناظر به حداکثر یک تیم نظیر می شود.

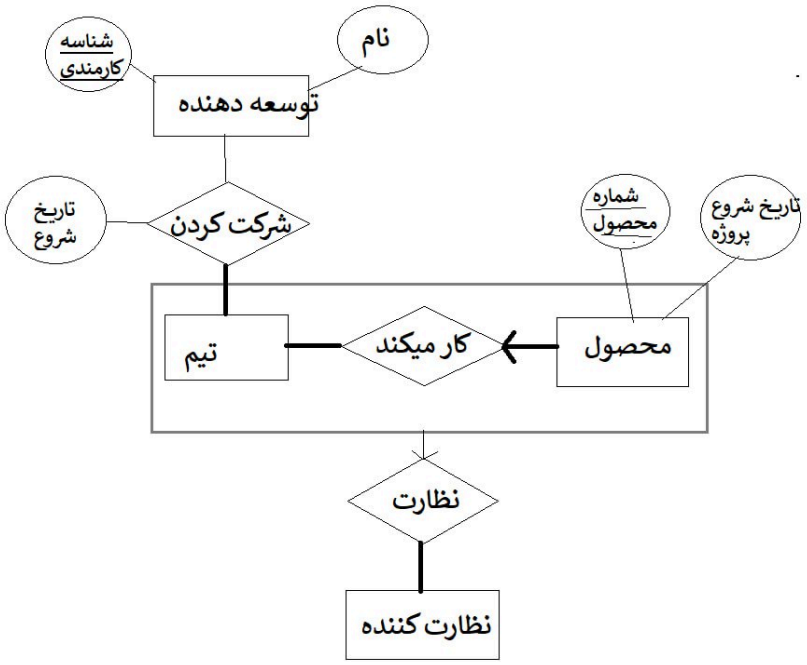
5. اگر توسعه دهنده امکان داشت در تیمی حاضر نباشد، رابطه توسعه دهنده با شرکت کردن به شکل یک خط معمولی رسم میشد.

6.یا سوابق درآمدزایی و فروش در ویژگی ذخیره میشود یا نه اگر بشود که مشکلی نیست اگر نشود، سوابق درآمدزایی و فروش محصول را در یک ویژگی ذخیره می کنیم که میتوان یک entity set مجزا برای آن داشت که هر محصول قابلیت چند درامد زایی را داشته باشد.

روش اول مطابق مفروضات فوق:


شناسه کارمندی
نام خانوادگی
نام
نوع قرارداد کارمندی
شناسه
تاریخ ایجاد پرونده
نظارت کردن
توسعه دهندگان
تیم
محصول
شرکت کردن
توسعه دادن
داشتن
تاریخ شروع به همکاری در تیم
سوابق
سوابق مدرنش
سوابق درآمدزای

روش دوم:

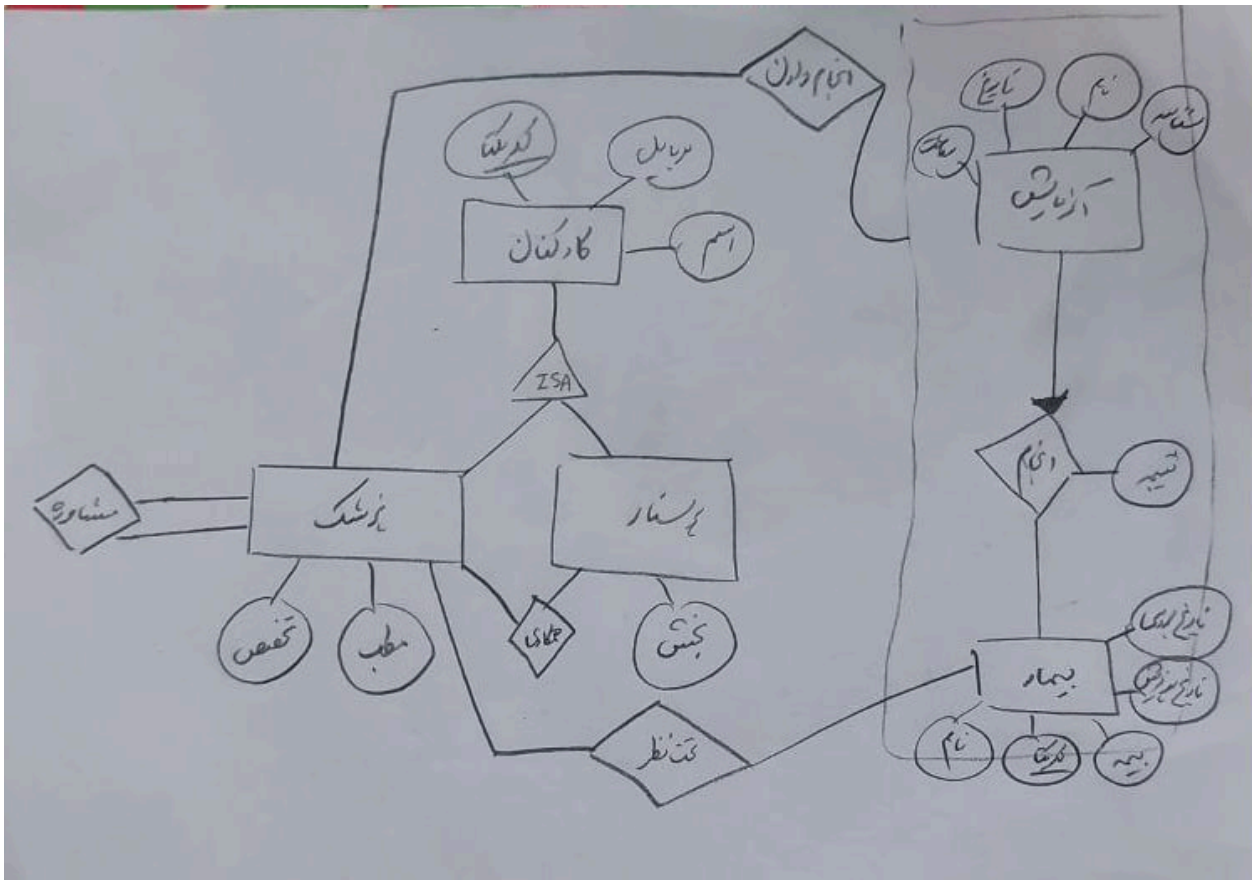


سوال 9)

مفروضات:

1. هر بیمار زیر نظر حداقل یک پزشک باید قرار بگیرد. بیمار بدون پزشک نداریم
2. هر آزمایش را حتما یک پزشک انجام داده
3. هر آزمایش برای دقیقا یک بیمار است
4. هر بیمار یکبار به بیمارستان مراجعه کرده. (اگر بیمار چند مراجعه به بیمارستان داشته باشد، تاریخ پذیرش را هم به عنوان primary key در نظر میگیریم. با این کار مشکل redundancy پیش می آید چون در هر مراجعه داریم نام و تاریخ بررسی و بیمه بیمار را دریافت می کنیم. راه حل تعریف موجودیت مراجعه است که دارای ویژگی های تاریخ پذیرش و تاریخ بررسی است و با بیمار رابطه دارد. در این حالت موجودیت بیمار فقط شامل بیمه و شماره یکتا و نام است)

نمودار:


انجام دادن
کد کلینیک
موبایل
کاردکتان
اسم
ZSA
پزشک
پرستار
مشاوره
تخصص
مطب
همکاری
بخش
تحت نظر
آزمایش
نام
تاریخ
شناسه
سفارش
انجام
تتبیه
بیمار
نام
کد ملی
بیمه
تاریخ آزمایش
تاریخ مراجعه